Appendix A: Comandi Bitcoin Explorer (bx)

```
Utilizzo: bx COMMAND [--help]

Info: I comandi bx sono:

address-decode
address-embed
address-encode
address-validate
base16-decode
base16-encode
base58-decode
base58-encode
base58check-decode
base58check-encode
base64-decode
base64-encode
bitcoin160
bitcoin256
btc-to-satoshi
ec-add
ec-add-secrets
ec-multiply
ec-multiply-secrets
ec-new
ec-to-address
ec-to-public
ec-to-wif
fetch-balance
fetch-header
fetch-height
fetch-history
fetch-stealth
fetch-tx
fetch-tx-index
hd-new
hd-private
hd-public
hd-to-address
hd-to-ec
hd-to-public
hd-to-wif
help
input-set
input-sign
input-validate
message-sign
```

```
message-validate
mnemonic-decode
mnemonic-encode
ripemd160
satoshi-to-btc
script-decode
script-encode
script-to-address
seed
send-tx
send-tx-node
send-tx-p2p
settings
sha160
sha256
sha512
stealth-decode
stealth-encode
stealth-public
stealth-secret
stealth-shared
tx-decode
tx-encode
uri-decode
uri-encode
validate-tx
watch-address
wif-to-ec
wif-to-public
wrap-decode
wrap-encode
```

Per maggiori informazioni, vedi la home page di Bitcoin Explorer e la documentazione per l'utente di Bitcoin Explorer.

Esempi d'uso del comando bx

Guardiamo a un po di esempi d'uso di Bitcoin Explorer per sperimentare con chiavi e indirizzi:

Genera un valore "seed" random usando il comando seed, che usa il generatore di numeri casuali del sistema operativo. Passa il seed al comando ec-new per generare una nuova chiave privata. Salviamo l'output (stdout) nel file *private_key*:

```
$ bx seed | bx ec-new > private_key
$ cat private_key
73096ed11ab9f1db6135857958ece7d73ea7c30862145bcc4bbc7649075de474
```

Ora, genera la chiave pubblica dalla chiave privata usando il comando ec-to-public. Passiamo il file *private_key* nello standard input e salviamo lo standard output del comando in un nuovo file

public_key:

```
$ bx ec-to-public < private_key > public_key
$ cat public_key
02fca46a6006a62dfdd2dbb2149359d0d97a04f430f12a7626dd409256c12be500
```

Possiamo cambiare formato alla public_key e farla diventare un indirizzo usando il comando ec-to-address. Passiamo la *public_key* come standard input:

```
$ bx ec-to-address < public_key
17re1S4Q8ZHyCP8Kw7xQad1Lr6XUzWUnkG
```

Le chiavi generate in questa maniera producono un wallet type-0 non-deterministico. Questo significa che ogni chiave è generata da un seed bitcoin indipendente. I comandi Bitcoin Explorer possono anche generare chiave deterministicamente, in conformità con il BIP0032. In questo caso, una chiave "master" è creata da un seed e successivamente estesa deterministicamente per produrre un albero di sotto-chiavi, risultante in un wallet deterministico type-2.

Per cominciare usiamo i comandi seed e hd-new per generare una to generate a una chiave master che verrà usata come base per derivare una gerarchia di chiavi.

```
$ bx seed > seed
$ cat seed
eb68ee9f3df6bd4441a9feadec179ff1

$ bx hd-new < seed > master
$ cat master
xprv9s21ZrQH143K2BEhMYpNQoUvAgiEjArAVaZaCTgsaGe6LsAnwubeiTcDzd23mAoyizm9cApe51gNfLMkBq
kYoWWMCRwzfuJk8RwF1SVEpAQ
```

A questo punto usiamo il comando hd-private per generare una chiave dell'"account" "hardened" e una sequenza di due chiavi private all'interno dell'account.

```
$ bx hd-private --hard < master > account
$ cat account
xprv9vkDLt81dTKjwHB8fsVB5QK8cGnzveChzSrtCfvu3aMWvQaThp59ueufuyQ8Qi3qpjk4aKsbmbfxwcgS8P
YbgoR2NWHeLyvg4DhoEE68A1n

$ bx hd-private --index 0 < account
xprv9xHfb6w1vX9xgZyPNXVgAhPxSsEkeRcPHEUV5iJcVEsuUEACvR3NRY3fpGhcnBiDbvG4LgndirDsia1e9F
3DWPkX7Tp1V1u97HKG1FJwUpU

$ bx hd-private --index 1 < account
xprv9xHfb6w1vX9xjc8XbN4GN86jzNAZ6xHEqYxzbLB4fzHFd6VqCLPGRZFsdjsuMVERadbgDbziCRJru9n6tz
EWrASVpEdrZrFidt1RDfn4yA3
```

Successivamente usiamo il comando hd-public per generare la sequenza corrispondente di due chiavi pubbliche.

```
$ bx hd-public --index 0 < account
xpub6BH1zcTuktiFu43rUZ2gXqLgzu5F3tLEeTQ5t6iE3aQtM2VMTxMcyLN9fYHiGhGpQe9QQYmqL2eYPFJ3ve
zHz5wzaSW4FiGrseNDR4LKqTy

$ bx hd-public --index 1 < account
xpub6BH1zcTuktiFx6CzhPbGjG3UYQ13WR16CmtbPiagEKpEVtpyjshWyMaMV1cn7nUPUkgQHPVXJVqsrA8xWb
GQDhohEcDFTEYMvYzwRD7Juf8
```

Le chiavi pubbliche possono essere derivate anche dalle chiave privati corrispondenti usando il comando hd-to-public.

```
$ bx hd-private --index 0 < account | bx hd-to-public
xpub6BH1zcTuktiFu43rUZ2gXqLgzu5F3tLEeTQ5t6iE3aQtM2VMTxMcyLN9fYHiGhGpQe9QQYmqL2eYPFJ3ve
zHz5wzaSW4FiGrseNDR4LKqTy

$ bx hd-private --index 1 < account | bx hd-to-public
xpub6BH1zcTuktiFx6CzhPbGjG3UYQ13WR16CmtbPiagEKpEVtpyjshWyMaMV1cn7nUPUkgQHPVXJVqsrA8xWb
GQDhohEcDFTEYMvYzwRD7Juf8
```

Possiamo generare un numero di chiavi praticamente senza limiti in una catena deterministica, tutte derivate da un singolo seed. Questa tecnica è usata in molte applicazioni wallet per generare chiavi che possono essere archiviate (backup) e ripristinate con un singolo valore seed. E' molto piu facile che dover effettuare il backup del wallet con tutte le chiavi generate randomicamente ogni volta che si crea una nuova chiave.

Il seed può essere codificato usando il comando mnemonic-encode.

```
$ bx hd-mnemonic < seed > words
adore repeat vision worst especially veil inch woman cast recall dwell appreciate
```

A questo punto il seed può essere decodificato usanto il comando mnemonic-decode.

```
$ bx mnemonic-decode < words
eb68ee9f3df6bd4441a9feadec179ff1
```

La codifica mnemonica può rendere il seed più facile da appuntare e anche da memorizzare.